DEUS XERCISES

Part I. The memory of the Deus X Machine has been loaded with instructions as shown on the following page. We have also given an assembly language version of the program, which may help you a bit in following it.

The input data consists of 5 lines in a file, as shown below; the fifth line is blank.

```
7
9
4
6
(blank line)
```

Please answer the following questions about the program that appears on the next page.

(a) Trace through the execution of this program. From time to time, a new value gets stored in register A. As you follow the execution of the program, write down, in order, each new value that gets stored in register A.

(b) What answer does this program print out?

(c) In one or two plain English sentences, describe what task this program performs. Your description should *not* just describe the operations ("First it puts zero into the A register ..."); rather, it should be something we could put in a software catalog so that a prospective buyer could know what the program does.

(d) There is one instruction which is not needed in this program. If it is removed the program will still accomplish its purpose (which you described in part **(c)**). Which instruction is it, and why don't you need it?

```
Address   Contents (for Part I)
          +-----------+
0.        |40|  |   0|                enta  0
          |--+--+--+--|
1.        |11|  |  33|                ldb   blank
          |--+--+--+--|
2.        | 5|  |  13|        read:   in    data
          |--+--+--+--|
3.        |51|  |  13|                cmpb  data
          |--+--+--+--|
4.        |60|  |  10|                je    print
          |--+--+--+--|
5.        |50|  |  13|                cmpa  data
          |--+--+--+--|
6.        |60|  |   2|                je    read
          |--+--+--+--|
7.        |62|  |   2|                jg    read
          |--+--+--+--|
8.        |10|  |  13|                lda   data
          |--+--+--+--|
9.        | 7|  |   2|                jump  read
          |--+--+--+--|
10.       |20|  |  34|        print:  sta   answer
          |--+--+--+--|
11.       | 6|  |  34|                out   answer
          |--+--+--+--|
12.       | 8|  |    |                halt
          |--+--+--+--|
13.       |  |  |  |  |        data:
          |--+--+--+--|
14.       |  |  |  |  |
          |--+--+--+--|
15.       |  |  |  |  |
          |--+--+--+--|
             . . .
          |--+--+--+--|
32.       |  |  |  |  |
          |--+--+--+--|
33.       |  |  |  |  |        blank:
          |--+--+--+--|
34.       |  |  |  |  |        answer:
          |--+--+--+--|
35.       |  |  |  |  |
          |--+--+--+--|
36.       |  |  |  |  |
          |--+--+--+--|
37.       |  |  |  |  |
          |--+--+--+--|
             . . .
```

Part II. In this problem, the memory has been reloaded with another program. The new memory contents appear as follows; this program's input is on the next page.

```
Address  Contents (for Part II)

          ------------
0.        | 5|  |  12|              in    count
          |--+--+--+--|
1.        | 5|  |  35|              in    name
          |--+--+--+--|
2.        |10|  |  12|              lda   count
          |--+--+--+--|
3.        |50|  |  10|    work:     cmpa  zero
          |--+--+--+--|
4.        |63|  |   9|              jl    done
          |--+--+--+--|
5.        |60|  |   9|              je    done
          |--+--+--+--|
6.        | 6|  |  32|              out   words
          |--+--+--+--|
7.        | 2|  |  11|              sub   one
          |--+--+--+--|
8.        | 7|  |   3|              jump  work
          |--+--+--+--|
9.        | 8|  |    |    done:     halt
          |--+--+--+--|
10.       |  |  |  | 0|    zero:         0
          |--+--+--+--|
11.       |  |  |  | 1|    one:          1
          |--+--+--+--|
12.       |  |  |  |  |    count:
          |--+--+--+--|
13.       |  |  |  |  |
          |--+--+--+--|
14.       |  |  |  |  |
          |--+--+--+--|
              . . .
          |--+--+--+--|
30.       |  |  |  |  |
          |--+--+--+--|
31.       |  |  |  |  |
          |--+--+--+--|
32.       | M  y     |    words:        My
          |--+--+--+--|
33.       | n  a  m e|                  name
          |--+--+--+--|
34.       | i  s  :  |                  is:
          |--+--+--+--|
35.       |  |  |  |  |    name:
          |--+--+--+--|
36.       |  |  |  |  |
          |--+--+--+--|
37.       |  |  |  |  |
          |--+--+--+--|
38.       |  |  |  |  |
          |--+--+--+--|
              . . .
          |--+--+--+--|
61.       |  |  |  |  |
          |--+--+--+--|
```

There are only two input lines for this program:

```
3
Sherlock
```

Please answer the following questions:

(a) What will the output from this program look like?

(b) Memory location 4 contains a branching instruction. Using the input lines and the program given above, will the branch in memory location 4 ever be taken?

(c) Under what circumstances *will* the branching instruction in location 4 be executed? (In other words, what has to happen in the program for that jump to be taken?)

(d) What could you put on the first input line that would cause the Deus X machine to execute the branch instruction in location 4?